COP 4610

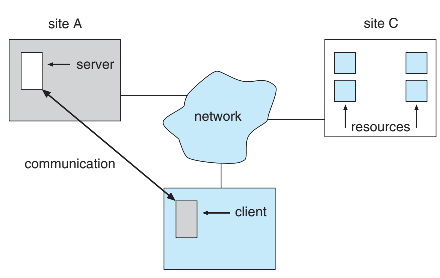
Operating System Principles

Distributed Systems & Networking Fundamentals

1

Overview

- A **distributed system** is collection of loosely coupled processors interconnected by a communications network
- Processors variously called ***nodes, computers, machines, hosts***
  - ***Site*** is location of the processor
  - Generally a ***server*** has a resource a ***client*** node at a different site wants to use



2

Reasons for Distributed Systems

- Reasons for distributed systems
  - **Resource sharing**
    - Sharing and printing files at remote sites
    - Processing information in a distributed database
    - Using remote specialized hardware devices
  - **Computation speedup** –
    - Load sharing or job migration
  - **Reliability**
    - Detect and recover from site failure, function transfer, reintegrate failed site

Types of Distributed Operating Systems

- Network Operating Systems
- Distributed Operating Systems

Network-Operating Systems

- Users are aware of multiplicity of machines
- Access to resources of various machines is done explicitly by:
  - Remote logging into the appropriate remote machine (telnet, ssh)
  - Remote Desktop (Microsoft Windows)
  - Transferring data from remote machines to local machines, via the File Transfer Protocol (FTP) mechanism
- Users must change paradigms – establish a **session**, give network-based commands
  - More difficult for users

Distributed-Operating Systems

- Users not aware of multiplicity of machines
  - Access to remote resources similar to access to local resources
- **Data Migration** – transfer data by transferring entire file, or transferring only those portions of the file necessary for the immediate task
- **Computation Migration** – transfer the computation, rather than the data, across the system
  - Via remote procedure calls (RPCs)
  - or via messaging system

Distributed-Operating Systems (Cont.)

- **Process Migration** – execute an entire process, or parts of it, at different sites
  - **Load balancing** – distribute processes across network to even the workload
  - **Computation speedup** – subprocesses can run concurrently on different sites
  - **Hardware preference** – process execution may require specialized processor
  - **Software preference** – required software may be available at only a particular site
  - **Data access** – run process remotely, rather than transfer all data locally

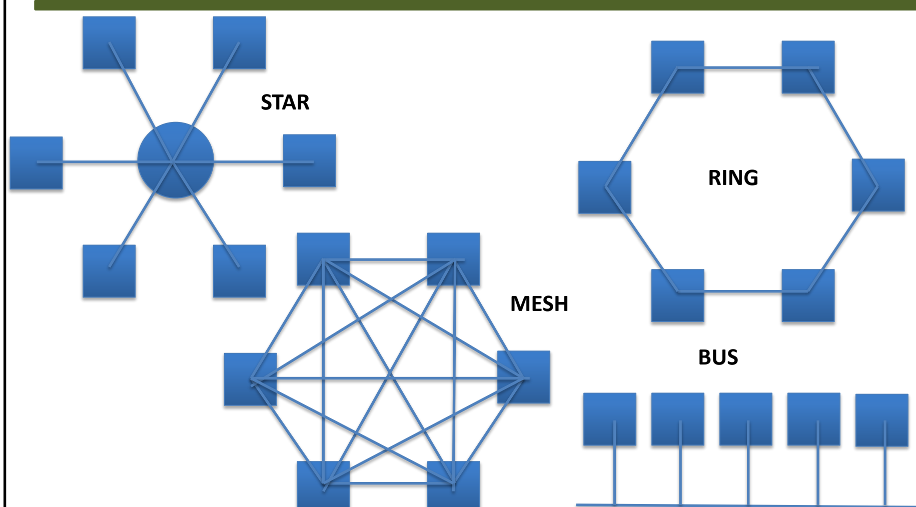
Connecting Computers

- **Local Area Network (LAN):** laboratory/office-scale
- **Metropolitan Area Network (MAN):** city-scale
- **Wide Area Network (WAN)**: world-wide (Internet -> "collection of networks")

Connecting Computers

- Sites in the system can be physically connected in a variety of ways; they are compared with respect to the following criteria:
  - **Installation cost** - How expensive is it to link the various sites in the system?
  - **Communication cost** - How long does it take to send a message from site *A* to site *B*?
  - **Reliability** - If a link or a site in the system fails, can the remaining sites still communicate with each other?
- The various topologies are depicted as graphs whose nodes correspond to sites
  - An edge from node *A* to node *B* corresponds to a direct connection between the two sites

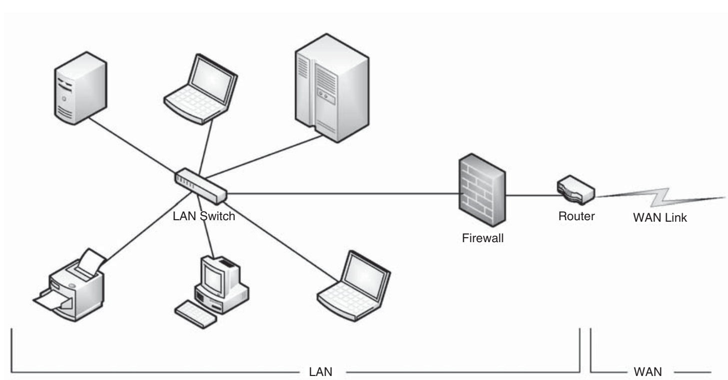
Network Topology



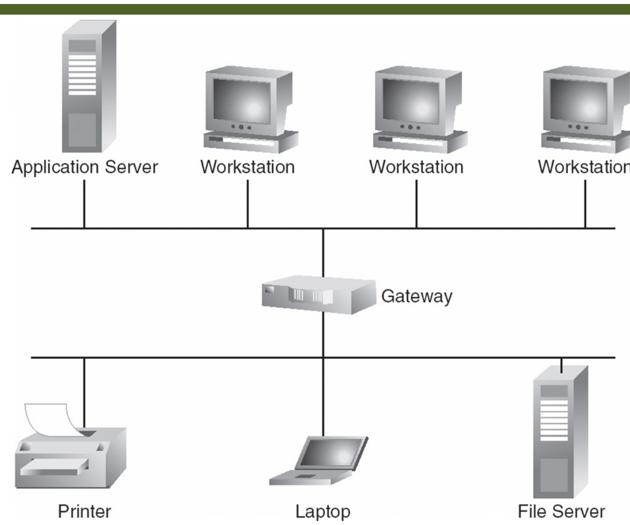
Examples

- **Ethernet:**
  - **popular, relatively inexpensive, easy-to-install LAN architecture**
  - uses the **CSMA/CD** media access control
  - data transmission normally occurs at 100 Mbps (10Mbps in the early forms and 10Gbps in the most recent forms)
  - partially described in the **IEEE 802.3** specification
- **Wi-Fi:**
  - **popular wireless LAN architecture**
  - uses a modified version of the **CSMA/CA** protocol
  - partially described in the **IEEE 802.11** specification

Local-Area Network

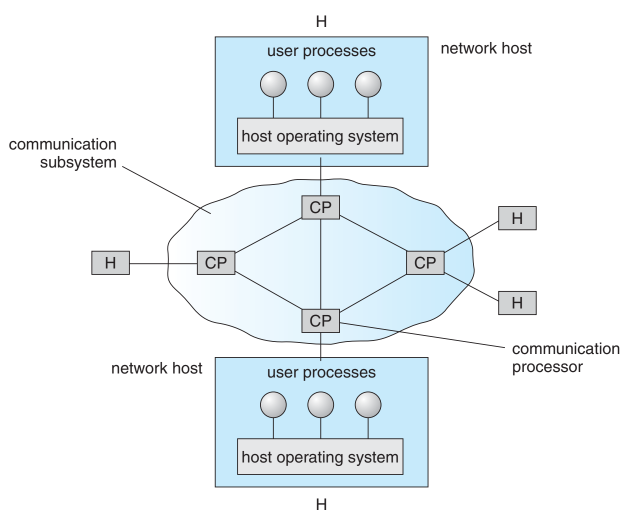


Local-Area Network



13

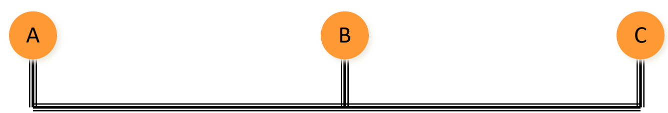
Wide-Area Network



14

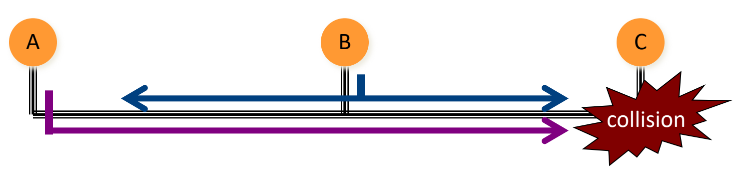
Medium Access Control (MAC)

- Responsible for deciding **when & how to transmit frames** over a network ("channel access problem")
  - Ethernet bus: computers connect to the same wire, i.e., two computers could "talk" at the same time: **collision**!
- MAC protocol is very important for "quality" of communications (successful transmissions, reliable transmissions, high throughput, low latency, fairness, …)



15

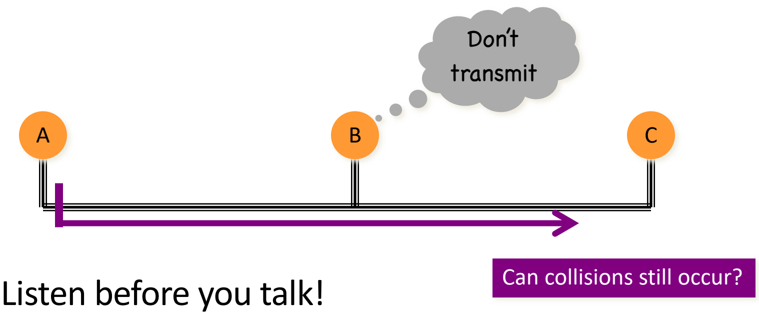
Simultaneous Transmissions



- Simple strategy:
  - Ignore ongoing communications and just transmit anytime:
    - large number of collisions
    - low throughput

16

"Smarter" Approach

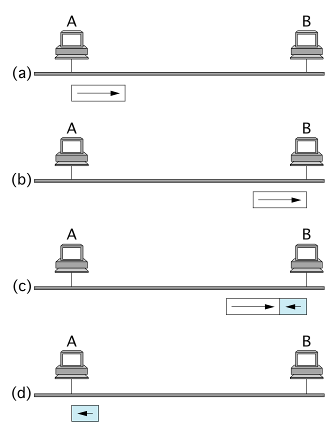


- Listen before you talk!
- **Carrier Sense Multiple Access (CSMA)**
  - "Sense" (listen) carrier ("is anyone else talking right now?")
  - If "busy": wait; if "idle": transmit

17

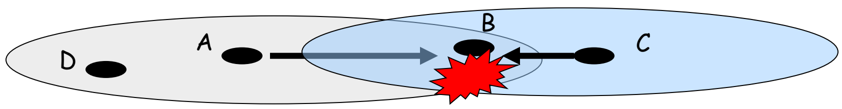
CSMA/CD

- **CD** = Collision Detection
- How? Keep listening to channel while transmitting!
- If transmitted signal and sensed signal differ:
  - Collision detected
  - Abort transmission
  - Jam channel: send random bit sequence to "inform" other computers



18

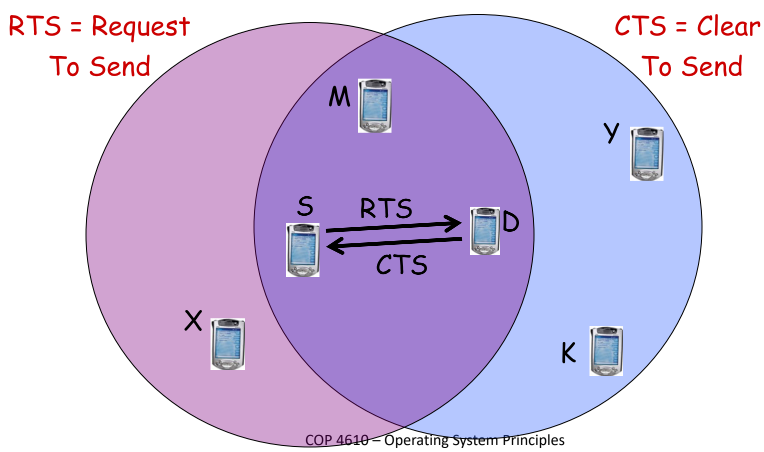
Collision Detection (Wireless)



19

IEEE 802.11 (CSMA/CA)

CA = Collision Avoidance



20

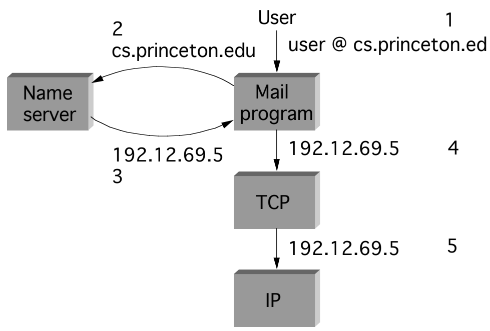
Communication Structure

- **Naming and name resolution** - How do two processes locate each other to communicate?
- **Routing strategies** - How are messages sent through the network?
- **Connection strategies** - How do two processes send a sequence of messages?
- **Contention -** The network is a shared resource, so how do we resolve conflicting demands for its use?

Naming and Name Resolution

- Name systems in the network
- Address messages with the process-id
- Identify processes on remote systems by
  <**host-name**, **identifier**> pair
- **Domain name system** (**DNS**) – specifies the naming structure of the hosts, as well as name to address **resolution** (Internet)

DNS



Routing Strategies

- **Fixed routing** - A path from *A* to *B* is specified in advance; path changes only if a hardware failure disables it
  - Since the shortest path is usually chosen, communication costs are minimized
  - Fixed routing cannot adapt to load changes
  - Ensures that messages will be delivered in the order in which they were sent
- **Virtual routing**- A path from *A* to *B* is fixed for the duration of one session. Different sessions involving messages from *A* to *B* may have different paths
  - Partial remedy to adapting to load changes
  - Ensures that messages will be delivered in the order in which they were sent

Routing Strategies

- **Dynamic routing** - The path used to send a message form site *A* to site *B* is chosen only when a message is sent
  - Usually a site sends a message to another site on the link least used at that particular time
  - Adapts to load changes by avoiding routing messages on heavily used path
  - Messages may arrive out of order
    - This problem can be remedied by appending a sequence number to each message
  - Most complex to set up
- Tradeoffs mean all methods are used
  - UNIX provides ability to mix fixed and dynamic
  - Hosts may have fixed routes and **gateways** connecting networks together may have dynamic routes

Routing Strategies

- **Router** is communications processor responsible for routing messages
- Must have at least 2 network connections
- Maybe special purpose or just function running on host
- Checks its tables to determine where destination host is, where to send messages
  - Static routing – table only changed manually
  - Dynamic routing – table changed via **routing protocol**

Routing Strategies

- Messages vary in length – simplified design breaks them into **packets** (or **frames**, or **datagrams**)
- **Connectionless message** is just one packet
  - Otherwise need a connection to get a multi-packet message from source to destination

Connection Strategies

- **Circuit switching** - A permanent physical link is established for the duration of the communication (i.e., telephone system)
- **Message switching** - A temporary link is established for the duration of one message transfer (i.e., post-office mailing system)
- **Packet switching** - Messages of variable length are divided into fixed-length packets which are sent to the destination
  - Each packet may take a different path through the network
  - The packets must be reassembled into messages as they arrive
- Circuit switching requires setup time, but incurs less overhead for shipping each message, and may waste network bandwidth
  - Message and packet switching require less setup time, but incur more overhead per message

ISO OSI Model

UPPER LAYERS	7	**Application Layer** ✓ Message format, Human-Machine Interfaces
UPPER LAYERS	6	**Presentation Layer** ✓ Coding into 1s and 0s; encryption, compression
UPPER LAYERS	5	**Session Layer** ✓ Authentication, permissions, session restoration
TRANSPORT SERVICE	4	**Transport Layer** ✓ End-to-end error control
TRANSPORT SERVICE	3	**Network Layer** ✓ Network addressing; routing or switching
TRANSPORT SERVICE	2	**Data Link Layer** ✓ Error detection, flow control on physical link
TRANSPORT SERVICE	1	**Physical Layer** ✓ Bit stream: physical medium, method of representing bits

Communication Protocol

- **Layer 1: Physical layer** – handles the mechanical and electrical details of the physical transmission of a bit stream
- **Layer 2: Data-link layer** – handles the *frames*, or fixed-length parts of packets, including any error detection and recovery that occurred in the physical layer
- **Layer 3: Network layer** – provides connections and routes packets in the communication network, including handling the address of outgoing packets, decoding the address of incoming packets, and maintaining routing information for proper response to changing load levels

Communication Protocol

- **Layer 4: Transport layer** – responsible for low-level network access and for message transfer between clients, including partitioning messages into packets, maintaining packet order, controlling flow, and generating physical addresses
- **Layer 5: Session layer** – implements sessions, or process-to-process communications protocols
- **Layer 6: Presentation layer** – resolves the differences in formats among the various sites in the network, including character conversions, and half duplex/full duplex (echoing)
- **Layer 7: Application layer** – interacts directly with the users, deals with file transfer, remote-login protocols and electronic mail, as well as schemas for distributed databases

What is IP, TCP, UDP?

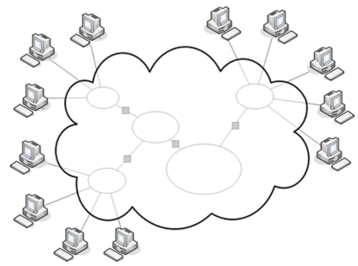
- Internet Protocol (IP):
  - Take your message and slap a “header” on it (“packet”)

Header	Data

  - What’s in a header?
    - Sender address: 112.44.44.23
    - Receiver address: 147.12.68.211
    - Routers use it to figure out what to do with it (see next slide for routers)
  - What does IP do:
    - mostly addressing
    - used by routers

Internet + Routers

- **Router: links parts of a larger network together**
- Routing using **tables**:
  - "129.74" belongs to University of Notre Dame
  - How is the table built?
    - Routers talk to each other to exchange what they know about the world (using ICMP = Internet Control Message Protocol)
  - Why only remember parts of a network?
    - 32-bit address consists of network address and computer address
    - Class A, B, C networks: 8/16/24 bits for network, rest for computers
      - Example: C network 127.45.20.21: 127.45.20 is network address, 21 is computer address (out of 255 computers)
- Routing is based on (independent) **packets**! (compare phone call vs. USPS)



UDP = User Datagram Protocol

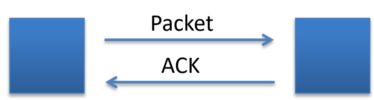
IP Header	UDP Header	Data

"IP Data"

- Slap on another header
- Adds more addressing: "**ports**"
  - IP address tell you which computer
  - Ports tell you which application on that computer
  - Example: a web server "listens" to requests on port 80
  - Web browser: http://www.google.com:80 = http://216.58.216.100:80
    - HTTP: HyperText Transfer Protocol
    - :80: optional
  - **Unreliable!**
    - Packets can get lost; packets can arrive out of order

TCP = Transmission Control Protocol

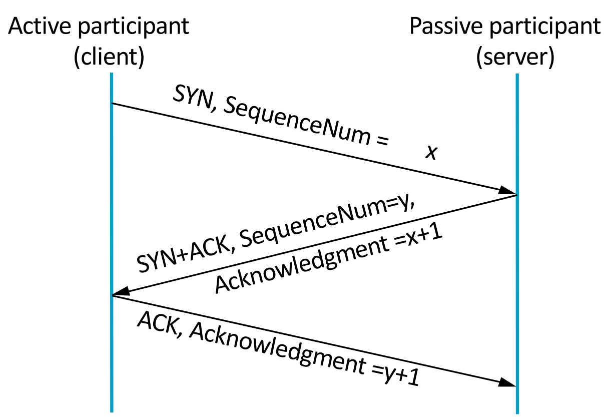
- **Reliable** protocol
- Adds ports (just like UDP), but also provides:
  - In-order delivery of packets (using sequence numbers)
  - Reliable delivery: using acknowledgment (ACK) packets



  - **Flow control:**
    - control of traffic between **sender and receiver**
    - receiver can throttle sender to avoid getting packets too fast
    - explicit: “advertised window” in ACK packet (how many more bytes)
  - **Congestion control:**
    - control of traffic flow into the **network**
    - routers can throttle sender to avoid getting too many packets
    - implicit: watch ACKs -> missing ACKs = router overload

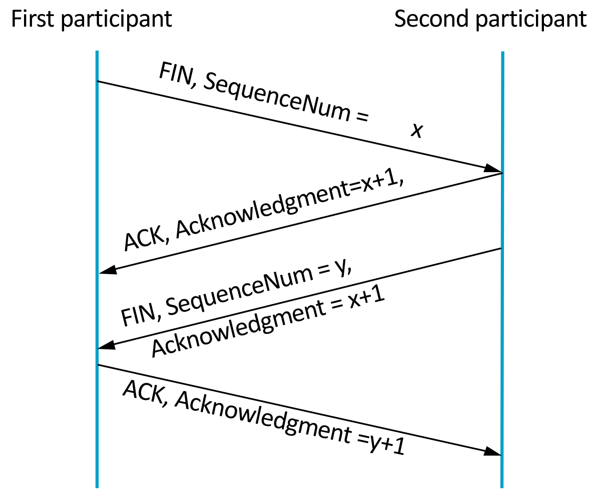
35

Connection Establishment



36

Connection Termination



UDP vs TCP

- TCP:
  - typical choice of most application
  - do not want to lose data, out-of-order arrival, etc.
  - email, web traffic, financial transactions, etc.

- UDP:
  - can be "faster"
    - no flow/congestion control "slowing down" traffic
    - no retransmissions
    - good for "real-time" traffic
  - out-of-order arrival: can also "reorder" at application level
  - loss of data: can be acceptable
    - missing frames in video/audio stream